Ministerstvo financií
Slovenskej republiky

EFQM
Recognised for excellence
5 star - 2012

Aplikačná bezpečnosť
Časť 3

Erik Saller, Ivan Oravec

KPMG
cutting through complexity™

Osnova

- Úvod
- Základné bezpečnostné hrozby pre aplikácie
- Aplikačné bezpečnostné funkcie
- Typické zraniteľnosti aplikácií a opatrenia proti nim
- Používanie otvorených štandardov
- Záver

Obmedzenie prístupu k aplikáciám

- Každá aplikácia môže obsahovať bezpečnostné nedostatky, ktoré dosiaľ neboli objavené, prípadne nie sú známe prevádzkovateľovi aplikácie.
- Z toho dôvodu je dobrou praxou obmedziť prístup k aplikácii len pre tých legitímnych používateľov a systémy, ktoré danú aplikáciu potrebujú pre svoju prácu.

Obmedzenie prístupu k aplikáciám (pokr.)

- Principiálne sa obmedzenie prístupu zabezpečuje hlavne:
  - Lokálne v rámci systému
  - Na sieťovej úrovni

Obmedzenie prístupu k aplikáciám lokálne v rámci systému (pokr.)

- Ak je možné aplikáciu spustiť lokálne na systéme jej používateľmi, je dobrou praxou aby bol prístup k takejto aplikácii pridelený iba jej legitímnym používateľom.
- Ak by prišlo ku kompromitácii neprivilegovaného používateľského účtu v systéme kde je nainštalovaná uvedená aplikácia, útočník nezíska priamy prístup k aplikácii ak daný účet nepatrí jednému z používateľov aplikácie.

Obmedzenie prístupu k aplikáciám lokálne v rámci systému (pokr.)

- V prípade, že v systéme nie sú všetci systémoví používatelia zároveň legitímnymi používateľmi uvedenej aplikácie, pomáha takéto opatrenie znížiť počet potenciálnych útočníkov.

Obmedzenie prístupu k aplikáciám lokálne v rámci systému na sieťovej úrovni

- Podobne, ako v prípade lokálneho prístupu, je dobrým opatrením obmedziť počet potenciálnych útočníkov aj na sieťovej úrovni.
- Ak je to praktické, najlepšie je toto opatrenie implementovať na úrovni siete, môžu byť však využité aj prostriedky OS, alebo samotnej aplikácie.
- Každopádne, obmedzenie prístupu k aplikácii pomáha znížiť riziko zneužitia neskôr objavenej zraniteľnosti aplikácie.

Obmedzenie prístupových práv aplikácie

- V informačnej bezpečnosti existuje tzv. princíp najmenších oprávnení (tiež známy ako zásada minimálnych oprávnení alebo zásada najnižšej autority), ktorý požaduje, aby na určitej vrstve abstrakcie výpočtového prostredia, mal každý modul (napr. proces, používateľ, program alebo jeho časť v závislosti na kontexte), prístup len k informáciám a zdrojom, ktoré sú nevyhnutné pre jeho legitímny účel.
- Znamená to, že napr. používateľský účet má mať pridelené iba tie oprávnenia, ktoré sú nevyhnutné pre prácu konkrétneho používateľa.

Obmedzenie prístupových práv aplikácie (pokr.)

- Napríklad používateľ pre zálohovanie nemusí inštalovať softvér, takže má mať práva iba na spustenie zálohovania a spustenie aplikácií súvisiacich so zálohovaním.
- Akékoľvek ďalšie privilégiá, napríklad pre inštaláciu nového softvéru, sú blokované.
- Táto zásada by mala platiť aj pre osobný počítač používateľa, ktorý zvyčajne pracuje na svojom normálnom používateľskom účte a na administrátorský účet chránený heslom sa prihlasuje (a následne pod ním pracuje) iba vtedy, keď si to situácia nevyhnutne vyžaduje.

Obmedzenie prístupových práv aplikácie (pokr.)

- Pri aplikácii na používateľa, sa používajú termíny ako „najmenší používateľský prístup“ alebo „najmenej privilegovaný používateľský účet (LUA)“.
- Použitie týchto termínov odkazuje na koncept, že všetky používateľské účty by v každom okamihu mali bežať s tak nízkymi oprávneniami ako je to len možné a tiež aplikácie by mali byť spúšťané s tak nízkymi oprávneniami ako je to len možné.
- Princíp najmenších oprávnení je široko akceptovaný dôležitý princíp používaný pri návrhu, ktorý výrazne prispieva k zvýšeniu ochrany dát, zníženiu dopadov zlyhaní aplikácie a obrane proti útokom.

Obmedzenie prístupových práv aplikácie (pokr.)

- Medzi výhody uplatnenia tohto princípu patria:
  - Lepšia stabilita systému
    - *Pokiaľ je kód aplikácie obmedzený čo sa týka zmien, ktoré môže v rámci systému vykonávať, je ľahšie testovať možné akcie a interakcie s inými aplikáciami, resp. modulmi tej istej aplikácie. V praxi napríklad aplikácie s obmedzenými právami nebudú mať dostatočný prístup na vykonávanie operácií, ktoré by mohli zhodiť systém alebo nepriaznivo ovplyvniť iné aplikácie spustené na rovnakom systéme.*
  - Lepšie zabezpečenie systému
    - *Pokiaľ je kód aplikácie obmedzený čo sa týka aktivít, ktoré môže v rámci systému vykonávať, zraniteľnosť v jednej aplikácii útočníkovi priamo neposkytne prístup ku zvyšku systému. Napríklad, Microsoft uvádza, že "Beh v štandardnom používateľskom režime poskytuje zákazníkom zvýšenú ochranu proti náhodnému narušeniu systému kvôli rôznym útokom, malware, spyware alebo nedetegovateľným vírusom.*

Obmedzenie prístupových práv aplikácie (pokr.)

- Jednoduchšie nasadenie
  - *Všeobecne platí, že čím menej oprávnení aplikácia vyžaduje tým je ľahšie jej nasadenie v rámci väčšieho prostredia. To obyčajne vyplýva z prvých dvoch výhod, aplikácie, ktoré inštalujú vlastné ovládače zariadení, alebo vyžadujú zvýšené bezpečnostné oprávnenia, zvyčajne v rámci svojho nasadzovania vyžadujú ďalšie úkony. Tak napríklad riešenia na platforme Microsoft Windows, ktoré nevyžadujú vlastné ovládače je možné spustiť priamo bez inštalácie, zatiaľ čo ovládače zariadení musia byť byť inštalované samostatne pomocou služby Windows installer, aby bolo možné poskytnúť ovládaču zvýšené oprávnenia.*
- Modularizácia (Kompartmentalizácia)
  - *Princíp najmenších oprávnení funguje oveľa lepšie, ak prístupová štruktúra aplikácie nie je typu "všetko alebo nič".*

Ministerstvo financií
Slovenskej republiky

EFQM
Recognised for excellence
5 star - 2012

Príklad č.1

- Povedzme, že idete na dovolenku a potrebujete ošetrovateľa pre svoje domáce zviera.
- Radi by ste obmedzili prístup ošetrovateľa len na vašu garáž, kde bude ponechaný váš domáci miláčik, kým ste preč.
- Ale ak nemáte garáž so samostatným zámkom, potom nemáte inú možnosť, len poskytnúť ošetrovateľovi kľúče ktoré mu umožnia prístup do celého domu.

KPMG
cutting through complexity™

Príklad č.1 (pokr.)

- Základnou myšlienkou modularizácie, že ak rozdelíme aplikáciu na čo najviac (v rozumnej miere) izolovaných jednotiek - modulov, môžeme zminimalizovať množstvo škôd, ktoré môže byť útokom na aplikáciu spôsobené.
- Rovnaký princíp sa uplatňuje pri ponorkách, ktoré sú vnútri rozdelené na mnoho rôznych komôr, z ktorých každá je samostatne hermeticky uzavretá.
- Ak porušenie trupu spôsobí naplnenie niektorej komory vodou, ostatné komory nie sú dotknuté.
- Zvyšok lodi tak môže zachovať svoju integritu a posádka môže prežiť v častiach ponorky, ktoré nie sú zaplavené.

Príklad č.2

- Ďalším častým príkladom princípu kompartmentalizácie je väzenie, kde je možnosť väčších skupín odsúdených zločincov zhromažďovať sa, minimalizovaná.
- Väzni nespia všetci spolu v jednej veľkej miestnosti, ale v menších celách po jednom alebo dvoch.
- Dokonca aj keď je im umožnené zhromaždiť sa, povedzme v jedálni, ďalšie bezpečnostné opatrenia sú adekvátne zvýšené aby pomohli kompenzovať značné zvýšenie rizika.

Príklad č.2 (pokr.)

- Vo svete IT, je oveľa jednoduchšie poukázať na príklady zlej modularizácie ako je nájsť tie dobré.
- Klasický príklad toho, ako to nerobiť, je štandardný unixový model privilégií, v ktorom sa z bezpečnostného hľadiska kritické operácie vykonávajú na báze "všetko alebo nič".
- Ak máte administratívne oprávnenie používateľa root, môžete v podstate robiť, čo chcete.

Príklad č.2 (pokr.)

- Ak nemáte root prístup, existujú rôzne obmedzenia.
- Bez root oprávnení napríklad nemôžete otvárať služby na portoch nižších než 1024.
- Rovnako tak nemôžete priamo pristupovať na mnoho zdrojov operačného systému.
- Zápis na disk nemôžete vykonávať napriamo, ale musíte ísť cez ovládače zariadení.

Príklad č.2 (pokr.)

- V súčasnej dobe, ak útočník úspešne zneužije chybu pretečenia zásobníka v kóde aplikácie ktorá beží pod administratívnym účtom, môže napr. vykonávať priamy zápis na disk a manipulovať s akýmikoľvek dátami v pamäti jadra operačného systému.
- Vo väčšine systémov nie sú žiadne ochranné mechanizmy, ktoré by tomu zabránili. Z toho dôvodu je napr. nemožné vytvoriť na lokálnom pevnom disku log súbor, ktorý nemôže byť útočníkom vymazaný.

Príklad č.2 (pokr.)

- Útočník bude mať v takomto prípade vždy možnosť obísť ľubovoľné inštalované ovládače, bez ohľadu na to, ako dobre tento ovládač sprostredkováva prístup na záznamové zariadenie.
- Preto je dôležité, aby aplikácie vo všeobecnosti nebežali pod administratívnym účtom, a ak existuje potreba týchto oprávnení pre výkon určitej časti funkčností, je dôležité aby boli tieto práva pridelené (po vhodnej modularizácii aplikácie) pridelené iba relevantnému modulu.
- Takisto ako veľa iných princípov, musí byť aj modularizácia použitá v rozumnej miere.
- Ak oddelíme každý menšiu funkčnosť do samostatného modulu, ľahko sa môže stať, že výsledný systém bude úplne nepraktický.

Uväznenie aplikácie (Jailing)

- Uväznenie aplikácie (Jailing) môžeme zjednodušene charakterizovať ako obmedzenie aplikačných účtov na konkrétny adresárový strom a vybrané príkazy.
- Na vytvorenie Jail prostredia v prostredí operačných systémov UNIX slúži systémové volanie chroot.
- Funkcia chroot v operačných systémoch UNIX vykoná operáciu, ktorá zmení zdanlivý koreňový adresár pre aktuálne bežiaci proces a jeho pod-procesy.
- Program, ktorý je spustený v takto upravenom prostredí sa nedokáže odkazovať (a preto zvyčajne nemá prístup) na súbory mimo určenej adresárovej štruktúry. Upravené prostredie sa nazýva "chroot jail" alebo jednoducho Jail.

Uväznenie aplikácie (Jailing) – pokr.

- Chroot prostredie možno použiť na vytvorenie a prevádzkovanie samostatnej virtualizovanej kópie softvérového systému. To môže byť užitočné okrem iného pre:
  - Testovanie a vývoj
  - Riadenie softvérových závislostí
  - Kompatibilita
  - Zotavenie systému
  - Oddelenie privilégií

Uväznenie aplikácie (Jailing) – pokr.

- Programy majú umožnené prenášať otvorené popisovače súborov (súbory, pipe štruktúry a sieťové pripojenia) do chroot prostredia, čo môže zjednodušiť návrh Jail prostredia tým spôsobom, že nie je nutné ponechávať pracovné súbory vnútri chroot adresára.
- To tiež zjednodušuje spušťanie potenciálne zraniteľných častí privilegovanej aplikácie v sandbox-e v rámci prevencie narušenia bezpečnosti.
- Treba poznamenať, že mechanizmus chroot nie je dostatočný na izoláciu procesu s administratívnymi (root) oprávneniami.

Uväznenie aplikácie (Jailing) – pokr.

- Chroot mechanizmus nie je určený k obrane proti úmyselnej manipulácii privilegovaným (root) používateľom.
- Na väčšine systémov sa chroot kontexty neskladajú správne a chroot programy s dostatočnými oprávneniami môžu vykonať druhú chroot operáciu a tak sa vymaniť z obmedzení.

Uväznenie aplikácie (Jailing) – pokr.

- Kvôli zníženiu rizika spojeného s touto bezpečnostnou slabinou, by sa chroot programy mali vzdať oprávnenia používateľa root akonáhle je to možné poprípade by sa mal použiť iný mechanizmus (ako napr. FreeBSD jail).
- Treba poznamenať, že niektoré systémy, ako napríklad FreeBSD, priamo implementujú preventívne opatrenia, aby sa zabránilo útoku druhou chroot operáciou.

Uväznenie aplikácie (Jailing) – pokr.

- Pri svojom spustení programy očakávajú, že nájdu dočasný odkladací priestor, konfiguračné súbory, uzly zariadení a zdieľané knižnice v určitých prednastavených adresároch.
- Aby sa chroot aplikácia úspešne spustila, musí byť chroot adresár vopred naplnený minimálnou množinou týchto súborov.

Uväznenie aplikácie (Jailing) – pokr.

- Iba administratívny používateľ root môže vykonať operáciu chroot.
- Toto má zabrániť používateľom v spustení SETUID programu vnútri špeciálne vytvoreného Jail prostredia (napríklad s falošným /etc /passwd a /etc/shadow súboru), ktoré by ho zmanipulovalo tak, aby používateľovi zvýšil oprávnenia.

Aplikačné firewally

- Aplikačný firewall je typ firewall-u, ktorý kontroluje vstup, výstup a/alebo prístup z, alebo na aplikácie alebo služby.
- Pracuje tak, že monitoruje a prípadne blokuje vstup, výstup alebo volanie systémovj služby, ak nespĺňa politiku nastavenú na firewall-e.
- Aplikačný firewall je zvyčajne schopný riadiť sieťovú prevádzku na ľubovoľnej vrstvy OSI až hore k aplikačnej vrstve.

Aplikačné firewally (pokr.)

- Je schopný kontrolovať aplikácie alebo konkrétne služby, na rozdiel od stavového sieťového firewall-u, ktorý bez dodatočného softvéru nie je schopný kontrolovať prevádzku v sieti s ohľadom na konkrétnu aplikáciu.
- Existujú dve základné kategórie aplikačných firewallov:
  - Sieťové aplikačné firewall-y,
  - Systémové (host-based) aplikačné firewall-y;

Sieťové aplikačné firewall-y

- Sieťový aplikačný firewall pracuje na aplikačnej vrstve OSI a je tiež známy ako proxy-based firewall alebo reverse-proxy firewall.
- Aplikačné firewall-y určené pre špecifický druh sieťovej prevádzky zvyknú byť pomenované podľa názvu konkrétnej služby, ako je napríklad web-aplikačný firewall.
- Sieťové aplikačné firewall-y môžu byť realizované prostredníctvom softvéru bežiaciho na hostiteľskom systéme alebo ako samostatný sieťový hardvér.

Sieťové aplikačné firewall-y (pokr.)

- Často ide o systém ktorý pomocou rôznych foriem proxy serverov podáva komunikáciu medzi klientom a serverom.
- Pretože pracuje na aplikačnej vrstve, môže kontrolovať obsah komunikácie a blokovať špecifický obsah ako sú určité webové stránky, vírusy alebo pokusy o zneužitie známych logických nedostatkov v klientskom softvéri.
- Moderné aplikačné firewally dokážu tiež odbremeniť servery od šifrovania, blokovať aplikačný vstup alebo výstup z detegovaných útokov alebo chybnú komunikáciu, riadiť alebo konsolidovať autentifikáciu prípadne blokovať obsah ktorý porušuje platné zásady.

Systémové (host-based) aplikačné firewall-y

- Systémový aplikačný firewall dokáže sledovať akýkoľvek aplikačný vstup, výstup alebo volania systémových služieb z aplikácií.
- Je to vykonávané tým spôsobom, že firewall monitoruje informácie podávané priamo v rámci systémových volaní namiesto získavania týchto informácií zo sieťovej komunikácie.
- Systémový aplikačný firewall môže poskytnúť ochranu len aplikáciám bežiacim na tom istom systéme.

Systémové (host-based) aplikačné firewall-y (pokr.)

- Aplikačný firewall vykonáva svoju funkciu tak, že rozhoduje, či by konkrétny proces mal akceptovať prichádzajúce sieťové spojenie.
- Systémový aplikačný firewall to rieši tým spôsobom, že zachytáva alebo aj filtruje volania medzi aplikačnou vrstvou a nižšími vrstvami OSI modelu.

Systémové (host-based) aplikačné firewall-y (pokr.)

- Príkladmi systémových aplikačných firewall-ov budúcej generácie, ktoré kontrolujú volania systémových služby sú AppArmor a TrustedBSD MAC rámec (sandboxing) implementovaný v operačnom systéme Mac OS X.
- Systémové aplikačné firewall-y môžu tiež poskytovať funkčnosť filtrácie sieťovej komunikácie.

Ministerstvo financií
Slovenskej republiky

EFQM
Recognised for excellence
5 star - 2012

POUŽÍVANIE OTVORENÝCH ŠTANDARDOV

KPMG
cutting through complexity™

Používanie otvorených štandardov

- Vytvárané alebo nakupované aplikácie, či vo všeobecnosti softvér ako taký, by mali byť postavené na otvorených štandardoch a nie na uzavretých proprietárnych, ktorých detaily nie sú verejne dostupné.
- Hlavné nevýhody aplikácií postavených na proprietárnych štandardoch:
  - Nemusia byť interoperabilné s existujúcimi alebo budúcimi produktmi iných výrobcov
  - Väčšinou nie je známa úroveň bezpečnosti týchto štandardov
  - Zákazník môže zostať v závislosti na službách a produktoch konkrétneho dodávateľa

Používanie otvorených štandardov (pokr.)

- Aby bol softvér vôbec užitočný, musí byť interoperabilný.
- Ak chceme vytlačiť dokument, program textového editoru musí byť schopný komunikovať s tlačiarňou.
- Podobne musí byť webový prehliadač schopný komunikovať s webovým serverom, atď.

Príklad

- Príklad s rozchod koľají je vhodná metafora, keď chceme pochopiť myšlienku otvorených štandardov. Rozchod koľají je vzdialenosť medzi koľajnicami na železnici.
- Ak by každé mesto alebo štát mal vlastný rozchod koľají alebo rozchody ich koľají by boli tajné, bolo by ťažké pre vlak prejsť od mesta k mestu.
- Dalo by sa to riešiť napríklad tak, že by bol podvozok rušňa a všetkých vagónov vlaku vymenený za iný na hranici každého mesta alebo štátu, alebo by musel byť skonštruovaný zložitý multi-prepravný systém umožňujúci prechod z jednej šírky do inej.

Príklad (pokr.)

- Alebo napríklad preprava tovaru z miesta na miesto, by vyžadovala vykladanie železničných vozňov na hraniciach a znovu naloženie nákladu do iného vlaku.
- Otvorený štandard pre softvér sa vzťahuje na verejne dostupné špecifikácie pre zvládnutie určitej úlohy.
- Jedná sa o súbor dohôd o niektorých aspektoch softvérového systému, ktorý sa týka kompatibility.

Príklad (pokr.)

- Najdôležitejšie otvorené štandardy sú dátové formáty. Napr. Web je závislý od noriem ISO pre znakové sady, a bez takého štandardu by obsahu Web-u mohlo byť porozumené len v malej oblasti, ktorá sa dohodla na znakovej sade. Servery a prehliadače by mohli byť použiteľné len v niektorých lokalitách.
- Otvorené štandardy predstavujú výhodu pre vývojárov, ale ešte viac sú prínosom pre používateľa tým, že umožňuje výber produktu a technológie.

Ministerstvo financií
Slovenskej republiky

EFQM
Recognised for excellence
5 star - 2012

Otázky?

KPMG
cutting through complexity™